Network Working Group
Request for Comments: 1091
Obsoletes: RFC 930

J. VanBokkelen
FTP Software, Inc.
February 1989

# Telnet Terminal-Type Option

## Status of This Memo

This RFC specifies a standard for the Internet community. Hosts on the Internet that exchange terminal type information within the Telnet protocol are expected to adopt and implement this standard.

This standard supersedes RFC 930. A change is made to permit cycling through a list of possible terminal types and selecting the most appropriate.

Distribution of this memo is unlimited.

## 1. Command Name and Code

TERMINAL-TYPE 24

## 2. Command Meanings

IAC WILL TERMINAL-TYPE

Sender is willing to send terminal type information in a subsequent sub-negotiation.

IAC WON'T TERMINAL-TYPE

Sender refuses to send terminal type information.

IAC DO TERMINAL-TYPE

Sender is willing to receive terminal type information in a subsequent sub-negotiation.

IAC DON'T TERMINAL-TYPE

Sender refuses to accept terminal type information.

IAC SB TERMINAL-TYPE SEND IAC SE

Server requests client to transmit his (the client's) next terminal type, and switch emulation modes (if more than one terminal type is supported). The code for SEND is 1. (See below.)

IAC SB TERMINAL-TYPE IS ... IAC SE

Client is stating the name of his current (or only) terminal type. The code for IS is 0. (See below.)

3. Default

WON'T TERMINAL-TYPE

Terminal type information will not be exchanged.

DON'T TERMINAL-TYPE

Terminal type information will not be exchanged.

4. Motivation for the Option

On most machines with bit-mapped displays (e.g., PCs and graphics workstations) a client terminal emulation program is used to simulate a conventional ASCII terminal. Most of these programs have multiple emulation modes, frequently with widely varying characteristics. Likewise, modern host system software and applications can deal with a variety of terminal types. What is needed is a means for the client to present a list of available terminal emulation modes to the server, from which the server can select the one it prefers (for arbitrary reasons). There is also need for a mechanism to change emulation modes during the course of a session, perhaps according to the needs of applications programs.

Existing terminal-type passing mechanisms within Telnet were not designed with multiple emulation modes in mind. While multiple names are allowed, they are assumed to be synonyms. Emulation mode changes are not defined, and the list of modes can only be scanned once.

This document defines a simple extension to the existing mechanisms, which meets both of the above criteria. It makes one assumption about the behaviour of implementations coded to the previous standard in order to obtain full backwards-compatibility.

## 5. Description of the Option

Willingness to exchange terminal-type information is agreed upon via conventional Telnet option negotiation. WILL and DO are used only to obtain and grant permission for future discussion. The actual exchange of status information occurs within option subcommands (IAC SB TERMINAL-TYPE...).

Once the two hosts have exchanged a WILL and a DO, the sender of the DO TERMINAL-TYPE (the server) is free to request type information. Only the server may send requests (IAC SB TERMINAL-TYPE SEND IAC SE) and only the client may transmit actual type information (within an IAC SB TERMINAL-TYPE IS ... IAC SE command). Terminal type information may not be sent spontaneously, but only in response to a request.

The terminal type information is an NVT ASCII string. Within this string, upper and lower case are considered equivalent. The complete list of valid terminal type names can be found in the latest "Assigned Numbers" RFC [4].

The transmission of terminal type information by the Telnet client in response to a query from the Telnet server implies that the client must simultaneously change emulation mode, unless the terminal type sent is a synonym of the preceding terminal type, or there are other prerequisites for entering the new regime (e.g., having agreed upon the Telnet binary option). The receipt of such information by the Telnet server does not imply any immediate change of processing. However, the information may be passed to a process, which may alter the data it sends to suit the particular characteristics of the terminal. For example, some operating systems have a terminal driver that accepts a code indicating the type of terminal being driven. Using the TERMINAL TYPE and BINARY options, a telnet server program on such a system could arrange to have terminals driven as if they were directly connected, including special functions not available to a standard Network Virtual Terminal.

Note that this specification is deliberately asymmetric. It is assumed that server operating systems and applications in general cannot change terminal types at arbitrary points in a session. Thus, the client may only send a new type (and potentially change emulation modes) when the server requests that it do so.

## 6. Implementation Issues

The "terminal type" information may be any NVT ASCII string meaningful to both ends of the negotiation. The list of terminal type names in "Assigned Numbers" is intended to minimize confusion

caused by alternative "spellings" of the terminal type. For example, confusion would arise if one party were to call a terminal "IBM3278-2" while the other called it "IBM-3278/2". There is no negative acknowledgement for a terminal type that is not understood, but certain other options (such as switching to BINARY mode) may be refused if a valid terminal type name has not been specified.

In some cases, either a particular terminal may be known by more than one name, for example a specific type and a more generic type, or the client may be a workstation with integrated display capable of emulating more than one kind of terminal. In such cases, the sender of the TERMINAL-TYPE IS command should reply to successive TERMINAL-TYPE SEND commands with the various names. In this way, a telnet server that does not understand the first response can prompt for alternatives. If different terminal emulations are supported by the client, the mode of the emulator must be changed to match the last type sent, unless the particular emulation has other Telnet options (e.g., BINARY) as prerequisites (in which case, the emulation will switch to the last type sent when the prerequisite is fulfilled). When types are synonyms, they should be sent in order from most to least specific.

When the server (the receiver of the TERMINAL-TYPE IS) receives the same response two consecutive times, this indicates the end of the list of available types. Similarly, the client should indicate it has sent all available names by repeating the last one sent. If an additional request is received, this indicates that the server (the sender of the IS) wishes to return to the top of the list of available types (probably to select the least of N evils).

Server implementations conforming to the previous standard will cease sending TERMINAL-TYPE SEND commands after receiving the same response two consecutive times, which will work according to the old standard. It is assumed that client implementations conforming to the previous standard will send the last type on the list in response to a third query (as well as the second). New-style servers must recognize this and not send more queries.

The type "UNKNOWN" should be used if the type of the terminal is unknown or unlikely to be recognized by the other party.

The complete and up-to-date list of terminal type names will be maintained in the "Assigned Numbers". The maximum length of a terminal type name is 40 characters.

## 7. User Interfaces

Telnet clients and servers conforming to this specification should

provide the following functions in their user interfaces:

Clients supporting multiple emulation modes should allow the user to specify which of the modes is preferred (which name is sent first), prior to connection establishment. The order of the names sent cannot be changed after the negotiation has begun. This initial mode will also become the default with servers which do not support TERMINAL TYPE.

Servers should store the current terminal type name and the list of available names in a manner such that they are accessible to both the user (via a keyboard command) and any applications which need the information. In addition, there should be a corresponding mechanism to request a change of terminal types, by initiating a series of SEND/IS sub-negotiations.

## 8. Examples

In this example, the server finds the first type acceptable.

Server: IAC DO TERMINAL-TYPE

Client: IAC WILL TERMINAL-TYPE

(Server may now request a terminal type at any time.)

Server: IAC SB TERMINAL-TYPE SEND IAC SE

Client: IAC SB TERMINAL-TYPE IS IBM-3278-2 IAC SE

In this example, the server requests additional terminal types, and accepts the second (and last on the client's list) type sent (RFC 930 compatible):

Server: IAC DO TERMINAL-TYPE

Client: IAC WILL TERMINAL-TYPE

(Server may now request a terminal type at any time.)

Server: IAC SB TERMINAL-TYPE SEND IAC SE

Client: IAC SB TERMINAL-TYPE IS ZENITH-H19 IAC SE

Server: IAC SB TERMINAL-TYPE SEND IAC SE

Client: IAC SB TERMINAL-TYPE IS UNKNOWN IAC SE

Server: IAC SB TERMINAL-TYPE SEND IAC SE

Client: IAC SB TERMINAL-TYPE IS UNKNOWN IAC SE

In this example, the server requests additional terminal types, and proceeds beyond the end-of-list, to select the first type offered by the client (new-type client and server):

Server: IAC DO TERMINAL-TYPE

Client: IAC WILL TERMINAL-TYPE

(Server may now request a terminal type at any time.)

Server: IAC SB TERMINAL-TYPE SEND IAC SE

Client: IAC SB TERMINAL-TYPE IS DEC-VT220 IAC SE

Server: IAC SB TERMINAL-TYPE SEND IAC SE

Client: IAC SB TERMINAL-TYPE IS DEC-VT100 IAC SE

Server: IAC SB TERMINAL-TYPE SEND IAC SE

Client: IAC SB TERMINAL-TYPE IS DEC-VT52 IAC SE

Server: IAC SB TERMINAL-TYPE SEND IAC SE

Client: IAC SB TERMINAL-TYPE IS DEC-VT52 IAC SE

Server: IAC SB TERMINAL-TYPE SEND IAC SE

Client: IAC SB TERMINAL-TYPE IS DEC-VT220 IAC SE

## 9. References:

[1] Postel, J., and J. Reynolds, "Telnet Protocol Specification", RFC 854, USC Information Sciences Institute, May 1983.

[2] Postel, J., and J. Reynolds, "Telnet Option Specification", RFC 855, USC Information Sciences Institute, May 1983.

[3] Solomon, M., and E. Wimmers, "Telnet Terminal Type Option", RFC 930, University of Wisconsin - Madison, January 1985.

[4] Reynolds, J., and J. Postel, "Assigned Numbers", RFC 1010, USC Information Sciences Institute, May 1987.

Reviser's note:

I owe much of this text to RFCs 884 and 930, by Marvin Solomon and Edward Wimmers of the University of Wisconsin - Madison, and I owe the idea of the extension to discussions on the "tn3270" mailing list in the Summer of 1987.

Author's Address

James VanBokkelen
FTP Software, Inc.
26 Princess Street
Wakefield, MA 01880-3004

Phone: (617) 246-0900

Email: jbvb@ftp.com